# BAB II

# LANDASAN TEORI

## 2.1 Tinjauan terhadap penelitian terkait.

Berikut beberapa penelitian terkait yang menjadi referensi pada penelitian ini: (Amrullah & Irawan, dkk., 2023) menurut jurnal yang saya kutip berjudul "Implementasi Jaringan Saraf Konvolusional dengan *Inception-V3* untuk Deteksi Katarak Menggunakan Gambar Digital Funduskopi" Dengan kemajuan teknologi, sektor pelayanan kesehatan saat ini menggabungkan peralatan medis dan teknologi informasi untuk meningkatkan mutu dan efisiensi layanan kesehatan. Sistem pelayanan kesehatan berbasis komputer menjadi alternatif di wilayah-wilayah yang memiliki keterbatasan sumber daya medis. Dampaknya adalah peningkatan kualitas layanan kesehatan.

(Rusdy Prasetyo, dkk., 2023) menurut jurnal yang berjudul "Jurnal Ilmiah Teknik Mesin, Elektro Dan Komputer Analisis Perbandingan Algoritma *Support Vector Machine (SVM)* Dan *Convolutional Neural Network (CNN)* Untuk Sistem Deteksi Katarak" Kebutaan yang disebabkan oleh katarak dapat disembuhkan melalui operasi dengan biaya yang terjangkau. Tingginya proporsi kebutaan yang disebabkan oleh katarak menunjukkan bahwa masih banyak penderita katarak yang belum menjalani operasi. Penyebab utama rendahnya angka operasi katarak di Indonesia meliputi berbagai faktor, salah satunya adalah kurangnya kesadaran masyarakat akan kondisi katarak yang mereka derita. Hal ini dipengaruhi oleh distribusi dokter mata yang tidak merata di seluruh provinsi Indonesia. Semakin lama seseorang menderita katarak atau menunda perawatan, semakin parah kerusakan pada penglihatannya akan terjadi.

## 2.2 Tinjauan Teori

### 2.2.1 Mata Katarak

Sel katarak adalah kondisi dimana lensa mata mengalami bercak putih yang menyerupai awan. Hal ini dapat mengganggu penglihatan mata, mempengaruhi jarak pandang, dan menyebabkan mata menjadi silau. Biasanya, katarak tidak menimbulkan iritasi atau rasa nyeri. Penderita katarak akan mengalami penglihatan yang kabur, mirip dengan melihat benda yang tertutup oleh kabut. (Guang-,dkk., 2017).

*Gambar 2.1 Contoh Mata Katarak*

Berikut adalah beberapa hal yang dapat menjelaskan lebih lanjut tentang penyakit mata katarak:

1. Struktur Mata: Mata manusia terdiri dari berbagai komponen, termasuk kornea (bagian depan yang transparan), iris (bagian berwarna yang mengatur cahaya masuk), lensa (bola transparan yang fokuskan cahaya pada retina), dan retina (lapisan dalam mata yang mendeteksi cahaya dan mengirimkan sinyal ke otak).
2. Fungsi Lensa Mata: Lensa mata berfungsi untuk mengatur fokus cahaya

pada retina, memungkinkan kita untuk melihat objek dengan jelas dan tajam.

3. Terbentuknya Katarak: Katarak terjadi ketika protein-protein di dalam lensa mata mulai berubah struktur atau menggumpal. Hal ini dapat disebabkan oleh faktor-faktor seperti penuaan, paparan sinar matahari berlebih, cedera mata, atau kondisi medis tertentu seperti diabetes.
4. Gejala Katarak: Beberapa gejala umum katarak termasuk penglihatan buram, sulit melihat dengan cahaya redup, sensitivitas terhadap cahaya terang, perubahan persepsi warna, dan sulit membaca atau melakukan tugas visual lainnya.
5. Jenis-Jenis Katarak: Ada beberapa jenis katarak, termasuk katarak senilis (yang terkait dengan penuaan), katarak kongenital (terjadi sejak lahir), katarak traumatis (akibat cedera mata), dan katarak sekunder (akibat dari penyakit atau kondisi medis lainnya).
6. Faktor Risiko: Beberapa faktor yang dapat meningkatkan risiko terjadinya katarak termasuk penuaan, paparan sinar matahari berlebih, merokok, konsumsi alkohol berlebihan, obesitas, dan riwayat keluarga dengan katarak.
7. Penanganan Katarak: Satu-satunya cara untuk mengatasi katarak adalah dengan operasi penggantian lensa. Prosedur ini melibatkan pengangkatan lensa yang mengalami kekeruhan dan penggantian dengan lensa buatan atau implan intraokular.
8. Pencegahan: Meskipun tidak selalu dapat dicegah sepenuhnya, ada beberapa langkah yang dapat diambil untuk mengurangi risiko terjadinya

katarak, seperti menghindari paparan sinar matahari berlebih, tidak merokok, dan menjaga kontrol kondisi medis seperti diabetes.

9. Komplikasi: Jika tidak diobati, katarak dapat menyebabkan penurunan penglihatan yang signifikan dan bahkan kebutaan. Oleh karena itu, penting untuk mendeteksi dan mengatasi katarak sejak dini.
10. Tinjauan Kesimpulan: Katarak adalah kondisi umum pada mata yang dapat mempengaruhi penglihatan seseorang. Dengan diagnosis dan penanganan yang tepat, penglihatan dapat membaik dan kehidupan dapat menjadi lebih baik. Jika Anda mengalami gejala katarak atau memiliki risiko untuk mengalaminya, jangan ragu untuk berkonsultasi dengan dokter mata.

### 2.2.2 *Machine Learning*

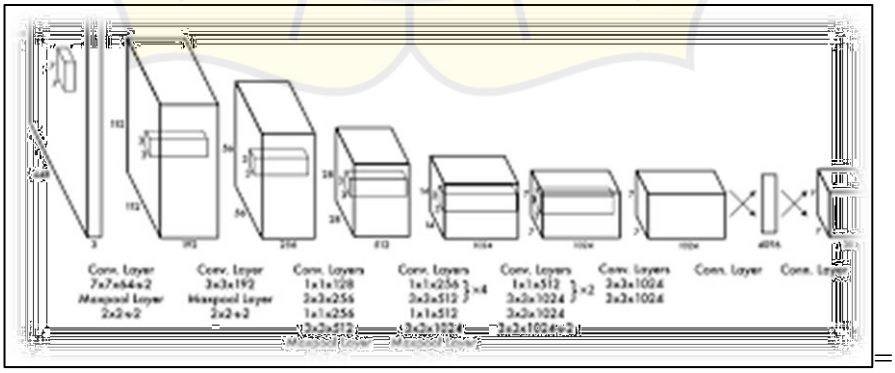
*Machine learning* merupakan wujud kecerdasan buatan yang memungkinkan system untuk belajar melalui data tanpa melalui pemrograman secara eksplisit (FACHRI RAMDHAN AL MUBAROQ, dkk., 2022). Dengan ini, dapat dibedakan secara mendasar bahwa algoritma machine learning dengan algoritma pemrograman umum.

Proses pembelajaran yang dimaksud mencakup upaya untuk meningkatkan kecerdasan melalui dua tahapan, yaitu latihan dan pengujian.(Huang dkk., 2006). Proses ini merupakan inti dari machine learning, di mana model terus memperbaiki kinerjanya seiring dengan pengalaman baru dan data tambahan yang diberikan kepadanya. Ini memungkinkan model untuk menjadi semakin akurat dalam membuat prediksi atau keputusan di masa yang akan datang.

### 2.2.3 Algoritma *You Only Look Once (YOLO)*

*YOLO* adalah metode *detektor* dengan model terpadu, di mana satu jaringan saraf tunggal dapat langsung memprediksi kotak pembatas dan *probabilitas* kelas dalam satu gambar penuh pada satu kali pemrosesan. Model *YOLO* dapat mengolah gambar masukan dengan kecepatan hingga 45 FPS, bahkan dengan versi jaringan saraf yang lebih kecil seperti *Fast YOLO* dapat mencapai 155 FPS, menjadikannya sebagai algoritma deteksi tercepat dibandingkan dengan metode deteksi real-time lainnya. Perbandingannya jauh lebih tinggi daripada metode *non-real-time* seperti *Fast R-CNN* dan *Faster R-CNN* yang hanya mencapai sekitar 0,5 FPS dan 7 FPS. Ini disebabkan oleh *YOLO* yang menerapkan teknik deteksi *single shot*, di mana *CNN* hanya dieksekusi sekali dalam proses deteksi objek. Ini berbeda dengan metode lain seperti *R-CNN* dan variasinya, yang menjalankan *CNN* beberapa kali untuk setiap proposal wilayah. (Maleh, dkk., 2023)

*YOLO* network terdiri dari 24 lapisan konvolusi yang diikuti oleh 2 lapisan terhubung penuh. Beberapa lapisan konvolusi menggunakan lapisan reduksi 1x1 sebagai alternatif untuk mengurangi kedalaman peta fitur, kemudian diikuti oleh lapisan konvolusi 3x3 seperti yang ditunjukkan pada gambar.



*Gambar 2.2 Arsitektur YOLO*

Terdapat tiga langkah dalam mendeteksi objek menggunakan *You Only*

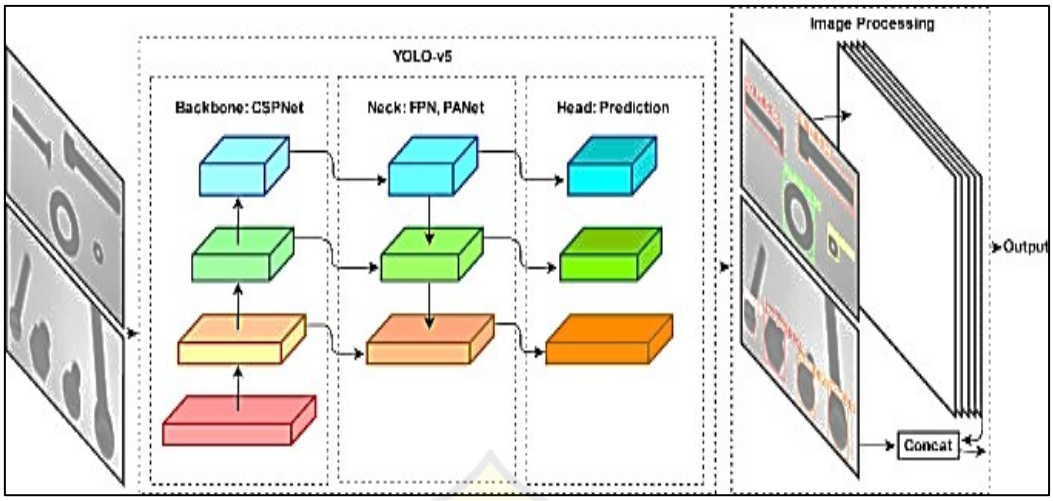
*Look Once* (*YOLO)* yang dilustrasikan pada Gambar 2.2 seperti berikut :

1. Mengubah ukuran masukan gambar menjadi 416 x 416.
2. Menjalankan syaraf tunggal *(single neural network)* pada gambar.
3. Melakukan *threshold* pada hasil deteksi berdasarkan nilai confidence.

*YOLO* berkinerja sangat baik dibandingkan dengan *single-shot detector* lainnya pada waktu itu dalam hal kecepatan dan akurasi. Ini bukan algoritma yang paling akurat dalam hal deteksi objek, tetapi tentu saja, ini menghasilkan kecepatan yang mengesankan dan dengan demikian merupakan keseimbangan yang baik antara kecepatan dan akurasi.

### 2.2.4 Arsitektur Algoritma *YOLOv5*

Struktur *YOLOv5* terbagi menjadi empat bagian utama: *input, backbone, neck, dan output*. Bagian input umumnya mencakup proses awal data, seperti *augmentasi* data *mosaic* dan pengisian gambar adaptif. Jaringan *backbone* menggunakan *Cross-Stage Partial Network (CSP)* dan *Spatial Pyramid Pooling (SPP)* untuk mengekstraksi peta fitur dari berbagai ukuran citra melalui serangkaian konvolusi dan pooling. sementara SPP digunakan untuk mengekstraksi fitur dari skala yang berbeda menjadi peta fitur. Ini dapat mempengaruhi akurasi model. Pada bagian *neck*, struktur *Feature Pyramid Network (FPN)* dan *Persistent Apparent Network (PAN)* digunakan. Kedua struktur ini dapat meningkatkan efisiensi dalam mengekstraksi fitur dari *backbone*. Pada bagian output, digunakan untuk memprediksi objek dari berbagai ukuran pada peta fitur. (Septyanto, dkk., 2022).




*Gambar 2.3 Arsitektur klasifikasi YoloV5*

Berikut adalah beberapa poin kunci mengenai arsitektur *YOLOv5*:

1. *Backbone*:

   *YOLOv5* menggunakan sebuah backbone *CNN* yang disesuaikan untuk tugas deteksi objek. Pada umumnya, arsitektur menggunakan model *backbone* seperti *CSPNet (Cross-Stage Partial Network)* atau *PANet (Path Aggregation Network)* untuk meningkatkan kapabilitas representasi fitur.

2. *Neck*:

   *YOLOv5* juga memasukkan elemen *neck* atau leher, yang bertanggung jawab untuk menggabungkan fitur-fitur dari berbagai tingkatan dalam jaringan. Ini membantu meningkatkan pemahaman kontekstual dan memberikan informasi yang lebih kaya kepada lapisan deteksi.

3. *Head*:

   *Head* dari *YOLOv5* terdiri dari lapisan-lapisan deteksi yang memprediksi lokasi dan kelas objek. Setiap lapisan deteksi menghasilkan

*bounding box* dan skor kepercayaan *(confidence score)* untuk setiap objek yang terdeteksi.

4. Prediksi Kepercayaan dan Kelas:

Setiap *bounding box* yang dihasilkan oleh model memiliki prediksi kepercayaan dan prediksi kelas untuk objek yang terdapat di dalamnya.

5. *Loss Function*:

*YOLOv5* menggunakan fungsi kerugian yang mencakup *loss* untuk lokalisasi dan *loss* klasifikasi. Hal ini membantu model untuk belajar secara efektif dan menghasilkan prediksi yang akurat.

6. Penggunaan Variasi Model:

Varian-varian model *YOLOv5*, seperti *YOLOv5s*, *YOLOv5m*, *YOLOv5l*, dan *YOLOv5x*, memberikan pilihan ukuran model yang berbeda, memungkinkan pengguna untuk memilih model yang sesuai dengan kebutuhan sumber daya dan akurasi.

7. Performa dan Kecepatan:

*YOLOv5* dikenal dengan performa tinggi dan kecepatan inferensi yang cepat, menjadikannya cocok untuk aplikasi real-time.

### 2.2.5 Tipe Algoritma *YOLOv5*

*YOLOv5 (You Only Look Once version 5)* adalah sebuah model deteksi objek dalam bidang visi komputer yang dirancang untuk memberikan deteksi objek secara cepat dan akurat. Tipe algoritma *YOLOv5* melibatkan penggunaan arsitektur *Convolutional Neural Network (CNN)* yang dikembangkan dengan

beberapa varian berdasarkan ukuran dan kompleksitas model. Berikut adalah tipe-tipe utama dari *YOLOv5*:

1. *YOLOv5s (Small)*:

   Versi kecil dari *YOLOv5* yang memiliki jumlah parameter yang lebih rendah, cocok untuk kasus penggunaan dengan keterbatasan sumber daya komputasi.

2. *YOLOv5m (Medium)*:

   Varian sedang yang menawarkan keseimbangan antara ukuran model dan kinerja deteksi objek. Cocok untuk berbagai aplikasi dengan kebutuhan komputasi sedang.

3. *YOLOv5l (Large)*:

   Varian besar yang memiliki lebih banyak parameter dan kapasitas untuk memahami representasi fitur yang lebih kompleks. Dapat memberikan performa lebih baik pada dataset yang lebih besar atau tugas yang lebih kompleks.

4. *YOLOv5x (Extra Large):*

   Varian sangat besar dengan jumlah parameter yang maksimal. Dirancang untuk tugas deteksi objek yang sangat kompleks atau memerlukan representasi fitur yang sangat mendalam.

   Setiap tipe *YOLOv5* memberikan *fleksibilitas* bagi pengguna untuk memilih model yang sesuai dengan kebutuhan spesifiknya, tergantung pada sumber daya komputasi yang tersedia dan tingkat akurasi yang diinginkan. Pemilihan tipe model dapat disesuaikan dengan kebutuhan tugas dan tingkat kompleksitas objek yang akan dideteksi.

*Table 2.1 Tipe-Tipe Algoritma YoloV5*

| Model | Size (pixels) | mAPval 0.5:095 | mAPval 0.5 | Speed CPU b1(ms) | Speed V100 b1(ms) | Speed V100 b32(ms) | Params (M) | Floops @640 (B) |
|---|---|---|---|---|---|---|---|---|
| YOLO v5n | 640 | 28.4 | 46.0 | 45 | 6.3 | 0.6 | 1.9 | 4.5 |
| YOLO v5s | 640 | 37.2 | 56.0 | 98 | 6.4 | 0.9 | 7.2 | 16.5 |
| YOLO v5m | 640 | 45.2 | 63.9 | 224 | 8.2 | 1.7 | 21.2 | 49.0 |
| YOLO v51 | 640 | 48.8 | 67.2 | 430 | 10.1 | 2.7 | 46.5 | 109.1 |
| YOLO v5x | 640 | 50.7 | 68.9 | 766 | 12.1 | 4.8 | 86.7 | 205.7 |

### 2.2.6 Bahasa Pemrograman *Python*

*Python*, sebuah bahasa pemrograman yang lahir pada tahun 1991 dan dikembangkan oleh Guido van Rossum, telah meraih ketenaran tinggi di dunia *teknologi*. Keunikan dan kelebihannya menjadikannya pilihan utama bagi para pengembang dalam berbagai *domain*, mulai dari pengembangan perangkat lunak hingga analisis data dan kecerdasan buatan.

*Python* merupakan bahasa pemrograman interpretatif yang serbaguna dengan filosofi desain yang menekankan pada keterbacaan *kode*. *Python* dianggap sebagai bahasa yang mengombinasikan kapabilitas dan kemampuan dengan sintaksis *kode* yang sangat mudah dipahami, serta dilengkapi dengan beragam *fungsionalitas* dalam pustaka standar yang luas dan komprehensif. Python juga didukung oleh komunitas yang besar. (Syahrudin & Kurniawan, dkk., 2020).

Bahasa ini tidak hanya populer di kalangan pemula, tetapi juga di kalangan peneliti tingkat dunia. Ketersediaan berbagai pustaka yang luas memungkinkan penyelesaian berbagai masalah dengan lebih mudah menggunakan bahasa ini.

### 2.2.7 *OpenCV*

*OpenCV*, singkatan dari "*Open Source Computer Vision*", merupakan sebuah program open source yang berbasis *C++* dan saat ini banyak digunakan dalam bidang *computer vision*. Program ini menyediakan lebih dari 500 fungsi untuk menangani visi komputer dan dapat diunduh dari situs web resminya di https://opencv.org/. *OpenCV* dikeluarkan dengan lisensi *BSD*, yang memungkinkan penggunaannya baik untuk keperluan bisnis maupun komersial. Sebagai *API library open source*, *OpenCV* sangat cocok untuk pemrosesan gambar. Platform yang didukung mencakup *Windows*, *Linux, Mac OS, iOS,* dan *Android*. *OpenCV* digunakan terutama untuk berbagai operasi gambar seperti membaca dan menulis gambar, deteksi wajah dan fiturnya, deteksi bentuk *geometris*, pengenalan teks dalam gambar, modifikasi warna dan kualitas gambar, serta pengembangan aplikasi *Augmented Reality*.

*OpenCV* terdiri dari lima *library* utama, yaitu *CV (algoritma pengolahan gambar dan vision), ML (library machine learning), Highgui (GUI, I/O gambar, dan video), CXCORE (struktur data, dukungan XML, dan fungsi grafis), dan CvAux (modul open source computer vision dari Intel Corporation).*

Beberapa keuntungan menggunakan *OpenCV* antara lain:

1. Tersedia banyak tutorial yang memudahkan pembelajaran.
2. Dapat digunakan hampir dalam semua bahasa pemrograman seperti *C++, Android SDK*, *JAVA*, *Python, dan C*.

3. Gratis untuk digunakan.

### 2.2.8 *Google Colabolatory*

*Google Colab*, secara lengkap dikenal sebagai *Colaboratory*, merupakan platform *cloud computing* yang disediakan oleh *Google*. Dirancang untuk memfasilitasi eksekusi kode *Python*, platform ini menawarkan lingkungan *notebook Jupyter* yang dapat diakses melalui *browser web*. Sebagai layanan gratis, *Colab* menyediakan sejumlah fitur yang sangat berguna untuk pengembangan, khususnya dalam konteks *machine learning* dan analisis data.

Dalam lingkungan *notebook Colab*, pengguna dapat menulis dan mengeksekusi kode *Python* secara interaktif dalam sel-sel terpisah. Kelebihan utama *Colab* adalah akses gratis ke unit pemrosesan grafis *(GPU)* dan pemrosesan tensor *(TPU)*, yang sangat bermanfaat dalam pelatihan model *machine learning* yang membutuhkan daya komputasi tinggi.

Integrasi dengan *Google Drive* menjadi fitur lain yang memudahkan pengguna untuk menyimpan dan berbagi *notebook* mereka. *Colab* juga mendukung berbagai pustaka dan *framework machine learning* populer seperti *TensorFlow dan PyTorch*, serta menyediakan kemampuan impor dan ekspor data yang mudah.

Selain itu, *Colab* menyediakan dokumentasi lengkap dan tutorial yang dapat diakses secara langsung dari lingkungan *notebook*. Ini membantu pengguna, terutama yang baru mengenal platform ini, untuk memahami dan memanfaatkan fitur-fiturnya dengan *optimal*. Kompatibilitas dengan format *Jupyter* memungkinkan pengguna untuk bekerja dengan notebook mereka di lingkungan lokal menggunakan aplikasi *Jupyter*.

*Google Colab* telah menjadi pilihan populer di kalangan pengembang dan peneliti yang mencari solusi *cloud computing* yang dapat diakses dengan mudah, efisien, dan secara gratis. Kombinasi fitur-fitur seperti akses ke *GPU/TPU*, integrasi dengan *Google Drive*, dan dukungan untuk pustaka *machine learning* membuatnya menjadi alat yang sangat berharga untuk eksperimen dan kolaborasi.

### 2.2.9 *Parameter Object Detection*

Evaluasi performa dalam tugas deteksi objek melibatkan beberapa parameter yang membantu mengukur sejauh mana model atau sistem dapat mengenali dan menentukan lokasi objek dalam sebuah gambar. Berikut adalah beberapa parameter evaluasi yang umum digunakan dalam tugas deteksi objek:

1. Presisi : Presisi mengukur seberapa banyak objek yang terdeteksi oleh model adalah objek yang sebenarnya. Presisi dihitung sebagai jumlah true positive dibagi oleh jumlah *true positive* dan *false positive*. Formula presis : $Presisi = \frac{True\ Positive}{True\ Positive + False\ Positive}$

2. *Sensitivitas* : *Sensitivitas* mengukur seberapa banyak objek yang sebenarnya berhasil dideteksi oleh model. Sensitivitas dihitung sebagai jumlah *true positive* dibagi oleh jumlah *true positive* dan *false negative*. Formula *sensitivitas* : $Sensitivitas = \frac{True\ Positive}{True\ Positive + False\ Negative}$

3. *F1-Score* : *F1-Score* adalah metrik gabungan yang menggabungkan presisi dan *sensitivitas*. *F1-Score* dihitung sebagai *2 × Presisi × Sensitivitas Presisi + Sensitivitas*. Metrik ini berguna ketika keseimbangan antara presisi dan sensitivitas diperlukan.

4. *IoU (Intersection over Union)* atau *Jaccard Index* : IoU mengukur sejauh

mana *bounding box* yang diprediksi oleh model tumpang tindih dengan *bounding box* yang sebenarnya. *IoU* dihitung sebagai area tumpang tindih antara kedua *bounding box* dibagi oleh area gabungan keduanya. Nilai *IoU* yang tinggi menunjukkan tingkat tumpang tindih yang baik antara prediksi dan *ground truth*.

5. *Metrik Average Precision (mAP)* : *mAP* adalah metrik yang umum digunakan dalam evaluasi deteksi objek yang melibatkan beberapa kelas. Ini menghitung rata-rata presisi pada beberapa nilai ambang *IoU*. *mAP* memberikan gambaran keseluruhan tentang seberapa baik model dapat mengenali objek di berbagai tingkat kesulitan.
6. *False Positive per Image (FPPI)* : *FPPI* mengukur jumlah *false positive* rata-rata per gambar. Ini memberikan pandangan tentang seberapa sering model membuat kesalahan dengan menghasilkan deteksi palsu pada setiap gambar.
7. *False Negative per Image (FNPI)* : *FNPI* mengukur jumlah *false negative* rata-rata per gambar. Ini menunjukkan seberapa sering model gagal mendeteksi objek yang seharusnya terdeteksi.
8. Waktu Komputasi : Parameter evaluasi ini mencakup waktu yang dibutuhkan oleh model untuk melakukan deteksi objek. Waktu komputasi menjadi penting terutama dalam implementasi di lingkungan *real-time* atau pada perangkat dengan sumber daya terbatas.

Pemilihan parameter evaluasi yang tepat tergantung pada kebutuhan spesifik dan tujuan dari tugas deteksi objek yang sedang dijalankan. Beberapa

parameter mungkin lebih penting tergantung pada konteks aplikasi dan preferensi pengguna.